Real-Time Imaging **4**, 263–274 (1998)
Article No.: ri970089

Real-Time Ultrasound Imaging Simulation

This paper presents a commercial real-time ultrasound simulator. An ultrasound volume is constructed from real ultrasound images in an off-line pre-process. Then simulated images are generated by slicing the volume in real-time. Such images can be computed very rapidly, including a post-processing enhancement. In most cases these synthetic images are indistinguishable from real ultrasound images.

This method needs to deal with the fact that an ultrasound image has view-dependent features and an acquisition-parameter-dependent character. This fact is two-fold: firstly, the pre-processed volume dataset includes some unwanted view-dependent features which should be removed. Secondly, the generated simulated image from a given arbitrary direction should be enhanced to include the appropriate view dependent features.

The paper describes the concepts of the ultrasound simulator, the real-time imaging algorithm and the off-line formation of the ultrasound volume dataset.

Dror Aiger[1,2] and Daniel Cohen-Or[1]

[1]*School of Mathematical Sciences,*
Tel-Aviv University, Ramat-Aviv 69978, Israel
[2]*MedSim Ltd., Kfar-Saba, Israel*

Introduction

Unlike other medical imaging tomography, ultrasound is a non-invasive and non-radiative device. The physician is free to scan the patient's internal organs and concentrate on locations of interest. Ultrasound imaging, however, is rather noisy, blurred and has low spatial and dynamic resolution. The physician has to gain considerable experience before he can diagnose.

This paper introduces UltraSim, a computer-based interactive simulator device designed to train physicians and technicians in medical diagnosis using ultrasound systems. Students using the system are able to learn how to identify and diagnose a wide range of medical cases by operating a simulated ultrasound machine on a mannequin, without any need for actual patients.

This application relates to the field of training personnel to work with expensive machines which require proficiency in eye–hand coordination; a field in which mistakes are very costly. These training systems enable trainees to practice the real life operation of complex systems without using the actual systems. Using the training systems saves expensive actual system time and provides training without putting anyone at risk.

An ultrasound examination consists of moving a transducer over the patient's body and scanning three-dimensional (3D) internal organs. It displays two-dimensional (2D) images of the slice of the examined organ according to the position and angle of the transducer. The 2D picture is then interpreted by the examiner. The examiner must use eye–hand coordination coupled with anatomic knowledge to find the best probe position and

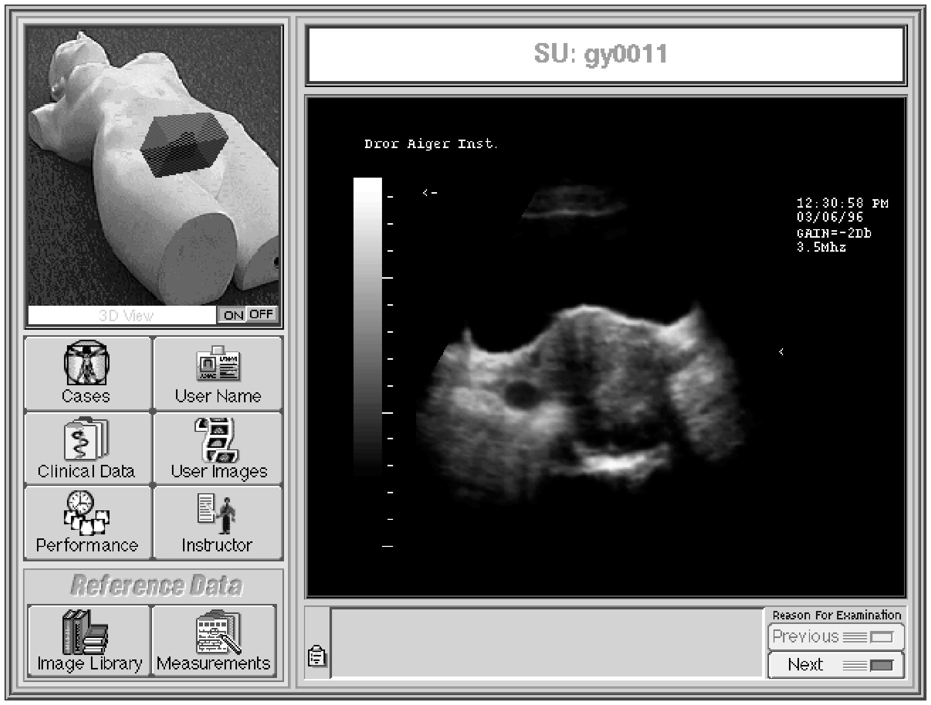

Figure 1. The simulator user interface, including a 3D view of the mannequin with the volume of interest.

angle in order to display the optimal picture for interpretation. Once the picture is found, the examiner can proceed to evaluate, compare and measure the data displayed and then determine the diagnosis of the case. Adjustment of ultrasound settings, questioning the patient and gathering additional data (such as previous examination results) also assists the examiner in determining the diagnosis.

Ultrasound training is currently done using several methods. One training method uses an actual ultrasound machine on other individuals. These are either fellow students or patients being examined. The main disadvantage of the system is the shortage of available patients and the inability to view predetermined pathologies. Patients with specific pathologies appear randomly, and depending on the rarity of the pathology it can take some time before a trainee actually encounters a specific pathology during training. Another training method involves using an actual ultrasound machine on 'phantoms'. These are physical devices containing substances which produce images in response to ultrasound waves. In this way the trainee can see the results of the 2D ultrasound display and practice interpreting the observed picture, using previous knowledge of the geometrical shapes in the phantom. Other training methods display the results of pre-recorded scans via video, textbook or computer media. None of the aforementioned training methods provide for training in the dynamic use of ultrasound on an artificial patient (mannequin). They do not allow for real-time change of ultrasound settings, selection of the correct transducer, and practice in the eye–hand coordination required to achieve the desired image.

A real-time, computer-based ultrasound simulator overcomes the drawbacks of other ultrasound training systems by providing real-time simulation of ultrasound images on a mannequin (Figure 1). The system simulates the function of an ultrasound machine including the probe, display functions and controls. The system enables a trainee to perform an ultrasound examination on an artificial patient. The simulated examination includes all stages usually followed during an actual ultrasound evaluation, such as identifying the anatomic area, patient history, correct control of the probe, display selection, image analysis and diagnosis. The advantage of this system is that it is interactive with true real-time imaging, providing the opportunity to explore the anatomy as if the patient and organs were actually present. There is no limited perspective or view, as in video- or computer-based training systems. The trainee can practice performing ultrasound examinations in conditions very similar to real life, learning the eye–hand coordination technique used during a real ultrasound examination. The simulator provides all the standard ultrasound functions, such as freeze, magnification, depth gain compensation,

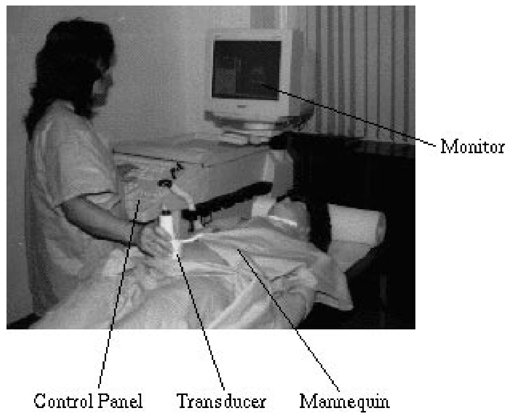

Figure 2. The UltraSim simulator.

zoom in and out, body marks, multi-image display and main gain.

In the following sections we present the simulator system, and an overview where we describe the characteristics of the ultrasound imaging and the general concept of the simulator. We present the real-time ultrasound image generation algorithm, and the off-line process of data acquisition, processing, registration and volume reconstruction is discussed.

The UltraSim System

The on-line training unit consists of a mock-up ultrasound system, a mock-up transducer and a mannequin (Figure 2). The system measures the position and angle of the transducer via a sensor installed in the mock-up transducer. It then displays in real-time the ultrasound image which was acquired off-line and which correlates with the current position and angle of the mock-up transducer on the mannequin.

Apart from the on-line training unit, the simulator consists of another part, the data collection unit, in which the ultrasound volumetric data are collected, off-line, from patients. The collected data are processed and stored in a large volumetric buffer. The data collection and generation of the volume buffer requires the solution of difficult registration problems of noisy ultrasonic data.

In addition to interactivity, the trainer is provided with a large library of cases, including rare pathologies, which can be taught during a relatively short period of time and as scheduled by the user or by the trainer. The system combines didactic instruction with exploratory learning via trial and error. With this structure specific lesson plans can be updated periodically, with new material allowing for the creation of a large encyclopedia of lessons using the same scanned data several times.

The training unit also provides background data on the patient, such as patient history, patient medical information and diagnostic data. In addition, the system provides the trainee with the option to view an ultrasound scan of the pathology as was performed by an expert. This feature gives the trainee a reference of comparison for his own scan. However, these and other didactic features are not in the scope of this paper, which focuses on the real-time imaging of ultrasound data.

The Principles of the Ultrasound Device

The ultrasound input device is the transducer which is manually positioned by the physician. The transducer converts electric energy into ultrasound energy and vice versa. It produces pulses of sound waves and sends them to the patient's body. It also receives the echos from the patient and converts them to electric energy. This energy is translated into an image that consists of gray level pixels which represent the structure of the body image in the ultrasound display. There are different kinds of transducers at different frequencies with which it is possible to determine the depth and the resolution of the image.

The physical principle of the ultrasound is as follows. Pulses of ultrasound, which are short pulses of sound wave at high frequency (1–15 MHz), are generated by the transducer (called pulse beam) and sent into the patient's body. They produce echos at organ boundaries and within tissues. These echos return to the transducer and are detected, processed and translated into appropriate gray level pixels which form the image on the ultrasound display. The gray level is a function of the reflection coefficient of the body at the appropriate location. The reflection coefficient is an attribute of the tissue depending on its physical, chemical and other characteristics.

The location of the pixels corresponds to the anatomic location of the echo-generating structure determined by knowing the direction of the pulse when it enters the patient and measuring the time for its echo to return to the transducer. From an assumed starting point on the display, the proper location for presenting the echo can then be derived, provided the direction in which to travel from that starting point to the appropriate distance is known. With knowledge of the speed of sound, the echo arrival time can be converted to distance to the structure that produces this echo. More details are provided in the next section.

Ultrasound Imaging

The core of ultrasound simulation is the capability of generating images in real-time that resemble real ultrasound images, including the typical ultrasound functions, depth gain compensation (DGC) and Gain. By real-time imaging we mean a frame rate of at least 10 Hz (over 10 frames per second). To generate an ultrasound image one can try to compute and simulate the physical mechanism of the ultrasound apparatus. However, it would be, nevertheless, extremely difficult to produce the very characteristic ultrasound images in real-time. A simple and apparently effective alternative is to form a volume from real ultrasound images in an off-line pre-process, and then slice the volume (on-line) to display a processed image of the oblique slice. Such images can be generated very rapidly, including post-processing enhancements, and can produce images which are, in most cases, indistinguishable from real ultrasound images (Figure 1).

However, this method of generating images from a pre-sampled ultrasound volume has some inherent problems, due to the fact that an ultrasound image has view-dependent features and an acquisition-parameter-dependent character. This fact is two-fold: firstly, the pre-processed volume dataset includes some unwanted view-dependent features that should be removed. Second, the generated simulated image from a given arbitrary direction should be enhanced and should include the appropriate view dependent features. These and other inherent problems are listed below.

Shadows

The ultrasound image exhibits shadows when closer objects obscure the sound waves from further objects. The shadows of a given image are correlated with the specific sampling direction. It is thus desirable to minimize this effect during data collection, because this feature is not reversible. The data at the shadow are lost and cannot be recovered unless the same area is sampled from a different viewing direction which views the shadow areas. The synthesis of shadows at the imaging phase is also easy to simulate in real-time. Our experience shows that the lack of shadows is not critical and the operator does not feel its absence.

Gain

The Gain control determines how much amplification is accomplished in the ultrasound receiver. Since the Gain operates on the image globally and has a uniform effect on the entire voltage received, it is not correlated with the specific sampling direction. The Gain is easily simulated, but problematic during data collection. If the data are sampled with too little Gain, weak echoes are not registered and these echos are lost. On the other hand, too much Gain causes saturation; that is, most echoes appear bright, and contrast resolution is lost. Gain effects the sampling volume in an irreversible manner, and it is thus important to sample with an appropriate Gain level.

Depth gain compensation

The DGC equalizes differences in received echo amplitudes as a function of the reflector depth. Reflectors at different depths with equal reflection coefficients produce different return amplitude echoes arriving at the transducer.

It is desirable to display echoes from similar reflectors in a similar way. The DGC functions as the Gain does, but at different levels as a function of the depth (the distance from the transducer). The user sets different Gain controls for different depths. Most ultrasound devices can set eight control points which define the DGC behavior. Like the Gain, the DGC is correlated with the sampling direction. During data collection, given the dependence on the sampling direction, the image should be as homogeneous and view-independent as possible.

The main problem with DGC and Gain is that they are irreversible, and some data are always lost during the collection and cannot be recovered from the sampled volume. However, with a good setup of the DGC and Gain levels it is possible to generate a volume buffer from which we get simulated images almost indistinguishable from real ultrasound images.

Focus

The width of the pulse beam generated by the transducer increases with depth, i.e. the beam has the shape of a cone whose apex is at the transducer. The pixel resolution is a function of the beam width. Thus, an ultrasound image exhibits varying resolutions at different depths.

The first problem is to simulate this different resolution based on one sampled volume taken with a specific machine and a specific transducer. Thus we need to use an ultrasound machine with a narrow beam to get an almost homogeneous sampled volume. In high end machines the beam size is small and we can neglect it in our simulation.

Very much like the operation of a camera and the physics of light that passes through lenses, the ultrasound beam can also be focused at an arbitrary field of view. It is possible to set the focus at an arbitrary depth and get the highest resolution at that depth.

The second problem related to focus is to simulate the arbitrary focal depth by changing the resolution at the related focal depth. One way to do this is to change the sample rate while generating the simulation image depending on the depth of the scan line (see the later section on '*Real-Time image generation*'.

Multiple focuses use a different pulse for each one of the required focuses, and the generated image has high resolution at several depths. However, using multiple focuses results in longer refresh rates. It is critical that the collection sampling time remains short to avoid the introduction of undesired movements. Thus, the volume is sampled in a single focus.

Resolution

The resolution of the acquired data is defined by the user who sets the *magnification* size. The acquired resolution is, of course, constant in the sense that it may be either over- or undersampled in the on-line process. During the collection phase the sampled resolution also affects the size of the entire image. If magnification is applied than we get a smaller area of higher resolution. This trade-off implies that acquiring data of higher resolution takes more time.

In most cases we do not sample at higher resolution and prefer to minimize the collection phase by sampling larger areas. However, in some cases certain pathologies are better learned from a smaller volume of higher resolution.

Another related problem is the uneven resolution of the sampled volume. The x–y slices (the sampled images) have a different resolution to the z–x planes (the inter-slice dimension). The shape of the ultrasound beam is not symmetric. It gets wider along the z axis than in the x–y plane (x–y is the ultrasound image plane). Thus, the x–y planes have a higher resolution than other planes. Our experience shows that this is not an aquatic problem and the resolution variations are hardly noticeable during the on-line simulation.

Noise

The ultrasound images are very blurred and noisy. However, this is not a real problem, since the simulation should retain these characteristics. They are also not view-dependent, and thus these attributes require no special treatment.

In summary, some of the above ultrasound features (DGC, Gain, Focus) can be alleviated by tuning down the acquisition parameters. However, it is not possible to remove them in a post-process. In the following section we describe the on-line imaging process in which all the above ultrasound features need to be simulated over the image with respect to the view direction and in accordance with the user's specific parameters.

Real-Time Image Generation

The common ultrasound imaging devices visualize a 2D cross-section of the 3D body. The cross-section is perpendicular to the probe and is of arbitrary orientation. The simulation of this image is basically a slicing algorithm of the volumetric buffer (Figure 3). The assumption is that the volume buffer is a large 3D array of voxels, each of which represents a view-independent *ultrasound sample*. The arbitrary slice is a virtual image frame buffer defined in a world coordinate system. The voxels pierced by the virtual frame are sampled, mapped and displayed in their image coordinate system after the frame is clipped against the volume buffer. In many cases the mapping between world and image space does not involve scaling, and the virtual frame can be *voxelized* with no filtering.

A voxelization algorithm for planar polygons has been proposed in [3]. The algorithm is basically an extension of the widely known 2D scan-line algorithm [6] where at each scan-line the third dimension is also interpolated. Later, Cohen and Kaufman [3] proposed a sweeping technique where a polygon can be generated by replicating one discrete line over the other and saving most of the

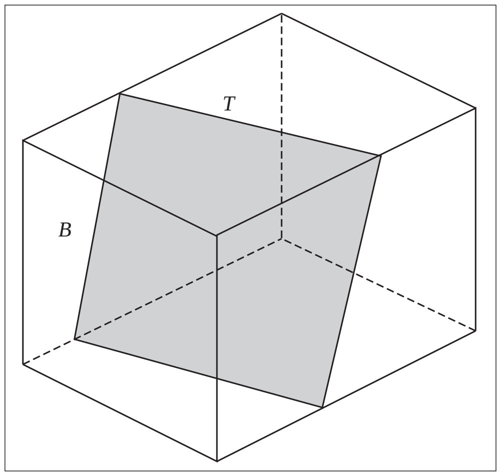

Figure 3. An oblique slice defined by sweeping a template line T along a base line B.

computations involved in the discretization process of the plane [4].

The sweeping technique is fast enough to voxelize slices in real-time on a Pentium processor (a 300×300 voxel slice can be scaled and displayed in a 400×400 pixel image in less than 1 s). However, when scaling is required, voxel oversampling and filtering are necessarily involved. Voxelization with scaling calls for the development of fast sampling algorithms of 3D polygons.

A brute force oversampling algorithm would use a trilinear interpolation of the dense lattice of sampling points. However, even an incremental computation of the trilinear function along a straight line would not avoid the excess of memory access, much of which is repeated.

This sampling process should incorporate the Gain and DGC functions, as well as other functions like Focus and casting shadows. The effect of the Gain and the DGC on the image is basically the same. The Gain affects the whole image simultaneously. The DGC function is set by eight potentiometers which control the effect on different areas of the image. The two values are combined to modify the pixel value as a function of its range in image space.

Given the four points which define the image frame in the world coordinates, the slicing algorithm scans all the voxels intersected by the frame and maps them to the image coordinate systems. The following algorithm is based on the weaving method in which the voxelized plane is generated by replicating a voxelized line, called a *template*, along a *base* voxelized line. Weaving is a natural technique for voxelizing surfaces which can be obtained by sweeping some fixed curve through space in such a way that the orientation of the swept curve remains unchanged throughout. The voxelized plane should be free of holes, which means a correct sample of the slice [2].

Weaving can be implemented very efficiently. If $\bar{T}_i = X_i, Y_i, Z_i$ are the coordinates of the ith voxel in the template T, then the sequence of offsets from a given reference point, say the template starting point ($\bar{T}_0$), is called the *template offset form*. The offset value $\bar{T}_i - \bar{T}_0$ is defined by:

$$dT_i = \bar{T}_i - \bar{T}_0 = Z_i * (sizeX * sizeY) + Y_i * sizeX + X_i \quad (1)$$

where $sizeX$ and $sizeY$ are the volume array dimensions. In other words, the δT value is the unfolded offset inside a linear volume.

The template offset form, denoted by T, is computed once at the outset and stored in an array. Then, for each voxel u in the base, denoted by B, the T array is used to incrementally construct a translate of the template, starting at u.

The basic algorithm that maps the voxels to the image buffer, $I[i][j]$. is a double loop which runs over all the u_j values of the base and all the v_i values of the template:

The inner loop runs over the i while j is constant. A pointer $ptr = \&(Volume[u_j])$ is used to further simplify the computation. A clipping process is necessary to avoid overflow of the $\bar{T}_i$, since Eqn 1 holds for $\bar{T}_i$ in the volume. This is done by clipping each row (template) against the volume so that the i index runs between the clipped boundary. The clipping cost is insignificant since it is applied only once in a row, while the memory access time to the voxels dominates the cost. Thus, it is most important to minimize the number of retrievals. Note that the preceding statement assumes that there is a one-to-one mapping between the voxel space and the pixel space, which is not necessarily true. The image space resolution is constant and defined by the display size. However, the slice size is defined on-line by the user, who can either zoom in or out.

For efficiency it is crucial to avoid unnecessary access to the volume buffer. The access time to the volume memory is dependent on the volume size, since adjacent voxels in large volumes have longer offsets, which tend to have a low cache hit ratio. In the case of a zoom out, the voxels are smaller than the pixels. Stepping inside the volume in a pixel-sized step is an undersampling of the voxels, and yields an image of reasonable quality. In the case of a zoom

```
for all $i$
    for all $j$
        $I[i][j] = Volume[u_j + v_i]$
```

Figure 4. The template-base algorithm that maps the voxels to the image buffer.

in, the voxels are oversampled and yield a displeasingly blocky and jaggy image.

To avoid the redundant oversampling of the same voxels, the zoom in image is generated by two interleaved processes. The first samples the volume, and generates an intermediate image whose pixels are of voxel size. The other process scales up the image into the final size. Since the image frame is scanned in scan line order, each intermediate row can be stretched before scanning the next row. Then the intermediate image needs to stretched vertically along its column to the final size. The scale process is decomposed into two series of one-dimensional stretches. This implies that the stretch function operates only on one-dimensional vectors and improves the efficiency. The stretch function must be fast enough to operate in real-time. In Figure 5 we present a pseudo-C code that stretches the values of array $A[m]$ into array $B[n]$ by piecewise linear interpolation. The scale factor is n/m and denoted by *Scale*. The stretch algorithm uses a similar concept to the line drawing algorithms which scan convert a line from 0,0 to n,m. Note that the slope is the inverse of the scale factor. The main loop takes n steps and the decision variable w is incremented by the slope. The value, w, serves as a weight for linear interpolation of the interval defined by two values of A, denoted by C_l and C_r:

$$c = C_r + ((C_r - C_l) * w) \tag{2}$$

Each time $w >= 1.0$ a new interval is interpolated. The interval endpoints C_l and C_r are updated and w is reset by a decrement of 1.0. The algorithm in Figure 5 is an implementation of the stretch algorithm using fixed point arithmetic where 1.0 is represented by 10 bits. Note that the stretch algorithm minimizes the access to the input and output buffers, since no buffer value is accessed twice.

A direct advantage of the row-by-row stretch principle is the simulation of the focus. As described in the previous section, the focus increases the image resolution at a given depth. It is possible to use the stretch function to augment or reduce the resolution of different rows to provide the user with the impression of higher resolution at a given depth, while rows out of focus depth are blurred by undersampling them and stretching them back to the image size.

Assume that the ultrasound has N potentiometers for the DGC function and one potentiometer for the Gain function. As explained previously, these two functions are essentially the same, and they amplify the signal reflected from the body according to the potentiometer's values. Thus, the value of the gray level of the voxel sampled at the volume buffer, denoted by V, is scaled by a scalar value *Gain*, to simulate the effect:

$$NewGray = MIN(Gain * V, 255) \tag{3}$$

This effect can be applied by simply modifying the color lookup table of the display system. For the DGC effect, the N values are interpolated by a spline to the length of the

```
Stretch(int *A, int *B, int m, int n)
{

    int *a = A, *b = B; /* point to the input and output buffers */
    float Scale = (float) n/(float) m; /* set the scale factor */
    int w = 0, dw = 1024/Scale; /* set the weights and their increments */
            /* w is a fixed point representation of the weight */
    int C_r = *a++, C_l = *a++; /* set the initial interval */
    while (n--) { /* loop over the output buffer B */
        *b++ = C_r + (((C_r - C_l) *w) >> 10;
        w+ = dw;
        if (w >= 1024) {
            w- = 1024;
            C_r = C_l;
            C_l = *a++;
            }
    }
}
```

Figure 5. The fixed point piecewise linear stretch algorithm.

image column, and stored in a lookup table $DGC[]$; so for row y:

$$NewGray = MIN(DGC[y] * V, 255) \tag{4}$$

Combining the effect of the two functions, we get:

$$NewGray = MIN(Gain * MIN(DGC[y] * V, 255), 255) \tag{5}$$

To save on computation, a 2D lookup table is used. The indices to the table are the gray values and the row number. Each table entry contains the gray level value to be displayed for a given sample value at a given depth for a preset Gain and DGC setup. This table needs to be updated each time some potentiometer's values are modified, or whenever the image is magnified.

Data Acquisition

The volume buffer which stores the ultrasonic data has to be big enough and represent a large portion of the mannequin to permit the bounded-free practice of a real life diagnosis. Contemporary ultrasound devices do not provide the capability of obtaining the entire volume in a single acquisition. This implies that the volume buffer has to be reconstructed from several subvolumes obtained from different viewpoints. The registration of monomodal datasets has been extensively investigated in medical application where atlas data are used [5]. However, ultrasonic datasets are far more problematic than other medical modalities, such as computed tomography (CT) or magnetic resonance imaging (MRI), since the ultrasound values are significantly more noisy, blurred and have many more view-dependent variations, as mentioned above. Moreover, the data sampled from a real patient is usually deformed, as will be explained below.

Given two volumes with a significant overlap, we want to find a spatial transformation which aligns and registers the two volumes into a single volume which smoothly combines the information from both. The type of registration technique that can be appropriately applied is directly dependent on the type of variation between the two volumes. Thus, to design a registration method it is necessary to know the type of variation exhibited by ultrasonic volumes. However, it is clear that volumes with large variations are more difficult to register, and we should look for "engineering solutions" that reduce possible variations in the first place.

The typical size of an ultrasound image generated by common ultrasonic devices is limited to 12–15 cm at the wide zone. The acquisition of a volume is thus reconstructed from a series of 2D slices. There are two main methods to collect the series of slices: a freehand collection and a mechanical collection.

In a freehand collection the location and orientation of the slice is tracked by a six-degree-of-freedom (6DOF) device (e.g. 6DOF, Isotrack). The slices are stored in the volume, and the gaps between the slices are filled by interpolations. This method seems impractical for large volumes because of the long acquisition time and the inaccuracy caused by the numerous unsampled gaps.

A better approach is to attach the transducer probe to a mechanical motor that sweeps the slice along some type of trajectory (e.g. fun, rotation). In particular, the TomTec device offers a parallel sweep by which a series of parallel uniformly spaced slices leave no gaps. It is possible to define the image resolution which is traded off for speed. The TomTec also includes three types of motors: parallel, fun and rotational, and gating equipment for periodic movements. The parallel dense slices generated by the TomTec provide small volumes of good quality. A series of such volumes needs to be collected and assembled to form a large volume.

The registration of two volumes requires one to detect the changes between the two images and to design a transformation that deforms them in order to remove or reduce the variations between them. The source variations can be classified into the following three types.

Directional variations

These variations are due to changes in the view point. They cause a misalignment that can be simply corrected by a rigid transformation. However, as we showed above, the acquisition of the same volume from a different view point causes other effects that are not compensated for by spatial transformation. For example, shadows are cast with strong correlation with the probe viewing direction.

Valumetric variations

These are caused by the characteristic of the ultrasonic technology. (For example, the DGC and Gain distortions and the inherent noisy and blurred ultrasound signal.) These effects are difficult to model and to remove. One can attempt to reduce them by tuning the acquisition parameters.

Geometric variations

Geometric deformations are caused by the movements of the body during the time of acquisition. Some movements are forced by the acquisition device, since the ultrasound probe must have good contact with the body. Of course the human body is soft and not flat, and it is rather difficult to maintain contact without causing forced movements by the muscles contracting. Immersing the body in a tube of water can avoid probe contact and eliminate the muscular contractions.

Another unavoidable deformation is caused by breathing and other natural behavior of the sampled body. Periodic deformation (like that of the heart) can be overcome by *gating*. In gating, the acquisition is synchronized with the period and the slices are acquired in the same phase of the period, using equipment similar to ECG, which monitors heart activity.

Volume Registration

Large ultrasound volumetric buffers are constructed using a series of volumes acquired by the TomTec. The TomTec is attached to a mechanical arm with which different volumes are obtained. The TomTec position and orientation are recorded using a 6DOF device with which the global misalignment can be corrected by a simple rigid transformation that maps the volumes back to a common (world coordinate) space. However, the global rigid transformation is coarse, and a fine elastic deformation is needed to obtain a good registration that compensates for local shape deformations and acquisition variations. The elastic deformation is local and is based on the overlapping portion of two given volumes.

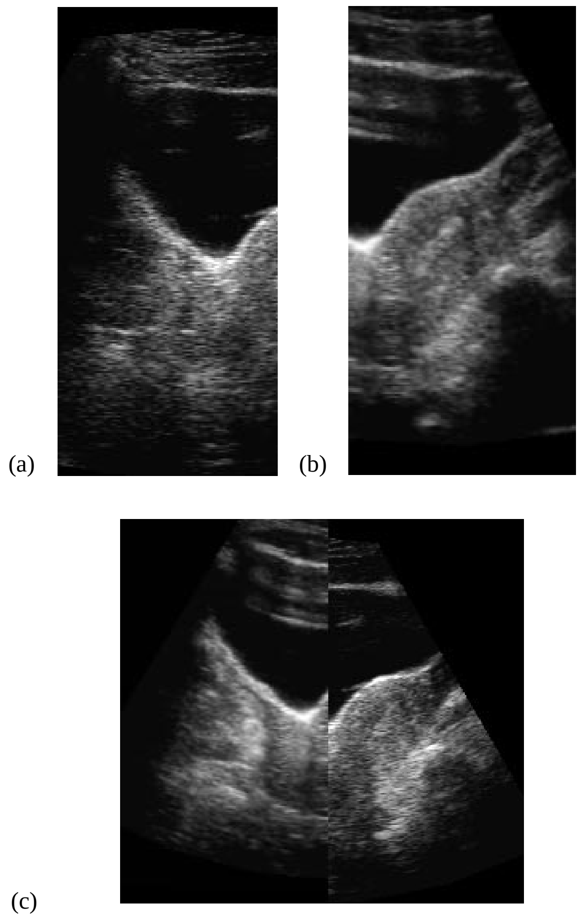

Figure 6. The registration of two volumes (Female Pelvis). (a) The first image; (b) the second image; (c) the two images combined by rigid alignment only.

The rigid transformation is too coarse and, even if exact, the two volumes have variations which are apparent mainly where the two volumes are in contact. See the two images in Figure 6(a) and 6(b) and the noticeably sharp transition in Figure 6(c). Applying a blending function to create a smooth transition can help, but since variations are due to misalignments or local deformations, an elastic warp is needed to adjust the two tangent slices. By applying the elastic to the tangent slices and falling off the warp deformation to the slices away, a better and smoother transition between the volumes is achieved (see Figure 7).

Barber [1] introduced a direct registration method to automatically correct small spatial variations caused by geometric deformations. The method is based on the gradient values, and has been successfully applied to a sequence of ultrasound images. The registration method we have developed is based on Barber's method. However, we have developed a multi-resolution method to better deal with large misalignments. The transformation is computed on a resolution pyramid and the results from the low resolution transformation are used to guide the computation of the finer levels. A resolution pyramid consists of the original image and a number of copies at lower resolutions.

At lower resolutions adjacent pixels and local gradients represent large distances of the original image. A displacement computed on a low resolution image indicates a larger displacement on the highest resolution of the original image. These larger displacements may yield transformations that compensate for larger misalignments. However, those are only rough transformations since they are based on coarse representations of the original images. The computation of the higher levels is based on the displacements

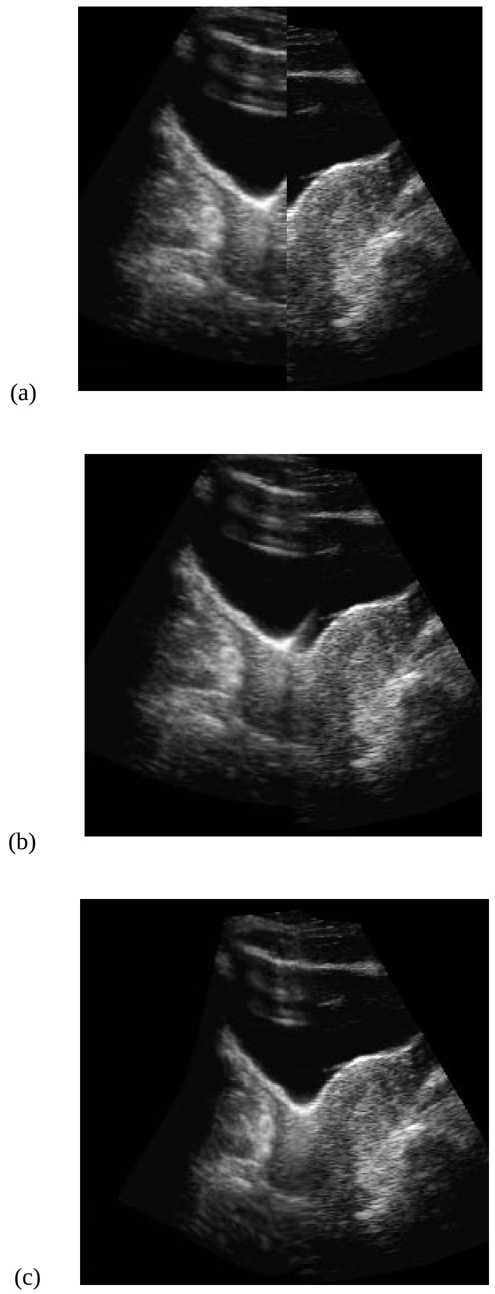

Figure 7. The registration of two volumes. (a) No blending and no warping. (b) Blending at the tangent slice. (c) Warping and blending.

of the lower levels and refines them. The multi-resolution method improves the performance of the registration in terms of the initial misalignment of the source and target images. The results presented in Figure 8 show that the single resolution method fails to register the two images, while the multi-resolution one quickly converges to a highly correlated registration.

We have been able to register volumes together (bound only by memory) to form a large volume buffer where the transitions among the subvolumes are not apparent (Figure 9). However, better results are expected by applying a true 3D elastic registration method, before gluing the sub-volumes together. We are currently extending the method to treat 3D volumes rather than 2D images. However, the 2D version is used to stitch two volumes together, as illustrated in Figures 6 and 7.

Conclusion

We have presented a novel application – an ultrasound simulator that operates in real-time on a Pentium PC. The computer-based simulator overcomes the drawbacks of other ultrasound training systems by providing real-time simulation of ultrasound images on a mannequin. The advantage of this system is that it is interactive with true real-time imaging, providing the opportunity to explore the anatomy as if the patient and organs were actually present.

The simulation is based on a simple and apparently effective concept where a data volume is constructed from real ultrasound images in an off-line pre-process. Then, in the on-line process, arbitrary slices of the volume are extracted and post-processed to simulate an ultrasound image. The synthetic image includes the typical ultrasound functions, such as DGC, Gain, Focus and Magnify. Future developments will include advanced ultrasound functions such as Doppler and Color.

The current development efforts are directed towards the off-line process. The construction of a large ultrasound volume is a great challenge, as was discussed in the section on data acquisition. We are currently developing a 3D registration technique which can better deal with inherent 3D deformations and correct larger misalignments. We are also developing an automatic partial matching algorithm to register two given volumes by a rigid transformation. We hope it will avoid the usage of 6DOF tracking devices. Another direction is the integration of 3D ultrasound imaging [7,8], since 3D visualization can help the trainee to understand the structure of the internal organs under consideration.

Acknowledgments

The authors wish to thank Professor Barry Goldberg and Mark Bergman for fruitful discussions. Special acknowledgments to Thomas Jefferson Ultrasound Research and Education Institute in Philadelphia for their help, and to Dr.

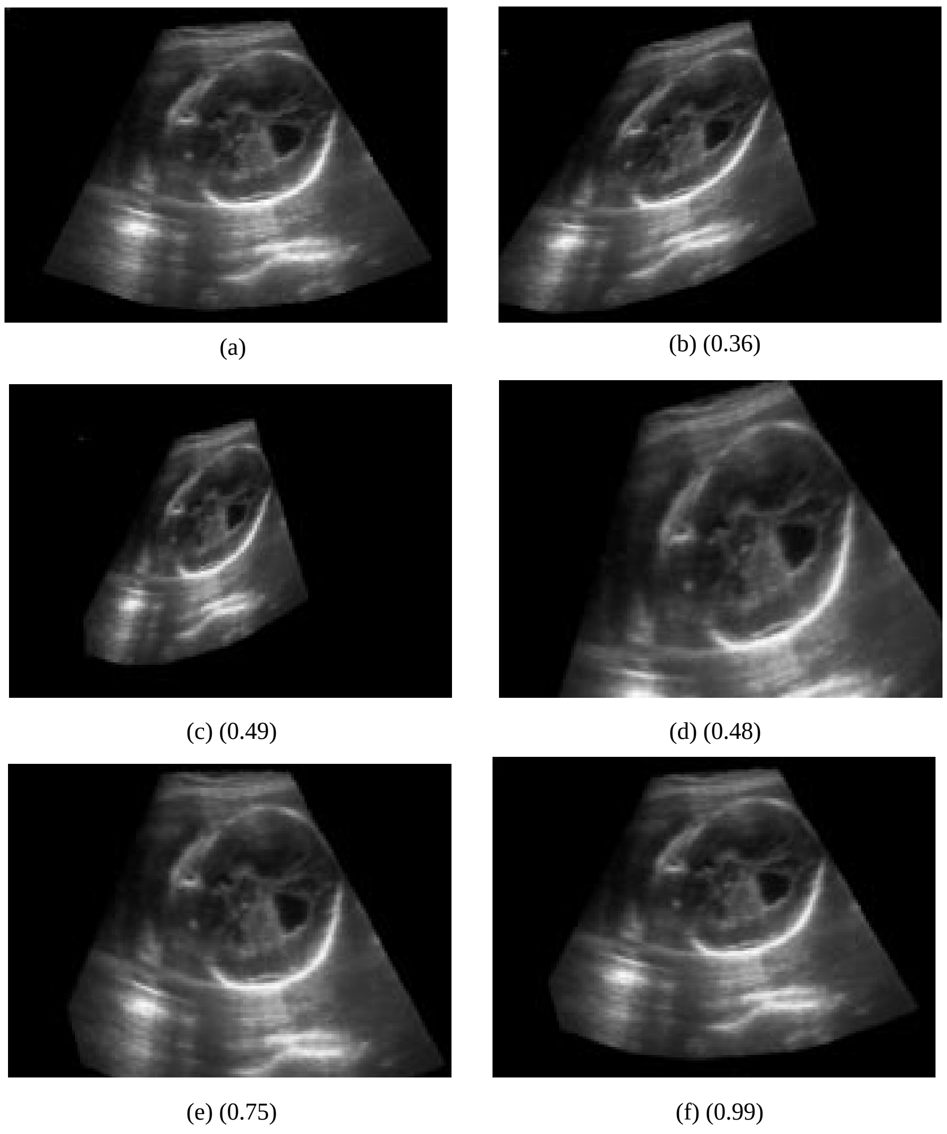

Figure 8. (a), (b) Multiresolution registration between two ultrasound images. (c) The registration using a single resolution. Using the multi-resolution technique; (d) after the first step at factor 48; (e) refined at factor 24; (f) refined at factor 6. The correlation values are in parentheses.

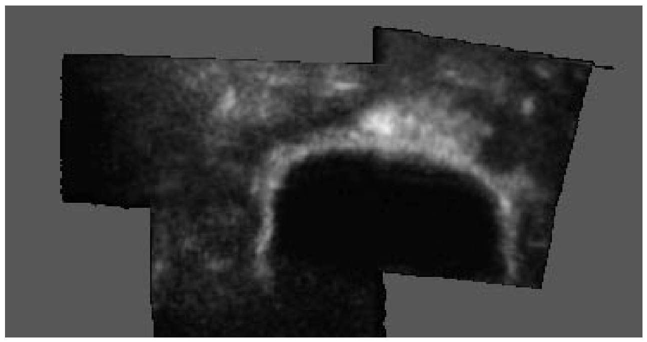

Figure 9. A cross-section of a volume constructed from six volumes.

Roni Tepper from Meir Medical Center in Israel for providing the ultrasound data.

The second author would like to acknowledge Georgios Sakas for helpful discussions and critiques of ultrasound simulation.

References

1. Barber, D.C. (1992) Registration of low resolution medical images. *Phys. Med. Biol.*, **37**: 1485–1498.

2. Cohen-Or, D. & Kaufman, A. (1995) Fundamentals of surface voxelization. *CVGIP: Graphics Models and Image Processing*, **57**: 453–461.
3. Cohen, D. & Kaufman, A. (1990) Scan conversion algorithms for linear and quadratic objects. In: *Volume Visualization*. Kaufman, A. (ed.) Los Alamitos, CA: IEEE Computer Society Press, pp. 280–301.
4. Cohen-Or, D., Kaufman, A. & Kong, T. Y. (1995) On the soundness of surface voxelizations. In: *Topological Algorithms for Digital Image Processing.* Yung Kong, T. and Rosenfeld, A. (eds.) North-Holland, Amsterdam, pp. 181–204.
5. Van den Elsen, P. A., Pol, E. D. & Viergever, M. A. (1993) Medical image matching – a review with classification. *IEEE Engineering in Medicine and Biology*, pp. 26–39.
6. Foley, J. D., van Dam, A., Feiner, K. & Hughes, F. (1990) In: *Computer Graphics Principles and Practice.* Addison-Wesley.
7. Nelson, T. & Elvis, T. (1993) Visualization of 3D Ultrasound Data. *IEEE Computer Graphics and Applications*, **13**: 50–57.
8. Sakas, G. & Walter, S. (1995) Extracting surfaces from fuzzy 3D ultrasonic data. In: *ACM Computer Graphics, SIGGRAPH-95*, Los Angeles, USA.